Optimizing Representations of Multiple Simultaneous Attributes for Gait Generation Using Deep Learning

Abhishek Sharma, *Member, IEEE*, and Eric Rombokas, *Member, IEEE*

***Abstract*— Rich variations in gait are generated according to several attributes of the individual and environment, such as age, athleticism, terrain, speed, personal "style", mood, etc. The effects of these attributes can be hard to quantify explicitly, but relatively straightforward to sample. We seek to generate gait that expresses these attributes, creating synthetic gait samples that exemplify a custom mix of attributes. This is difficult to perform manually, and generally restricted to simple, human-interpretable and handcrafted rules. In this manuscript, we present neural network architectures to learn representations of hard to quantify attributes from data, and generate gait trajectories by composing multiple desirable attributes. We demonstrate this method for the two most commonly desired attribute classes: individual style and walking speed. We show that two methods, cost function design and latent space regularization, can be used individually or combined. We also show two uses of machine learning classifiers that recognize individuals and speeds. Firstly, they can be used as quantitative measures of success; if a synthetic gait fools a classifier, then it is considered to be a good example of that class. Secondly, we show that classifiers can be used in the latent space regularizations and cost functions to improve training beyond a typical squared-error cost.**

***Index Terms*— Resentation learning, autoencoders, generative models, multi-task learning, style transfer, assistive devices, exoskeletons, personalization.**

I. Introduction

MANY applications in computer vision and robotics require generating novel human movement profiles. One such application is control of an assistive exoskeleton, for which we might need reference trajectories that vary according to desired speed, terrain, and the personal style of the user. References [1], [2], [3], and [4] are some previous approaches to producing trajectories. Some have included multiple attributes, for example speed and ramp terrain [1] and [4]. These methods use careful selection of basis functions, and optimize the error w.r.t. the training samples. This works well if a limited number of attributes are to be used to generate trajectories, and more importantly, when the attributes can be easily represented. However, with increasing availability of multi-modal datasets, there is a need for general-purpose methods that generate gait based on multiple attributes e.g., speed, personal style, terrain, mood etc. (Fig. 1a) There are also no general methods for prioritizing different attributes or to introduce new ones. In addition to reducing the need for hand-crafted features, these methods also need to maintain a degree of interpretability, which is essential for clinical applications.

In other domains, eg. computer vision and audio synthesis, the notion of *style transfer* has been introduced, where the idiosyncrasies or fundamental attributes of one example are combined with those of another to generate a completely novel sample. Fig. 2 shows an example of this, for the MNIST dataset, where digits are generated with desired handwriting styles. We adapt the method used for handwriting style transfer [5], for personalized gait generation at desired speeds, and show that can be made to work for this application domain. However, it doesn't allow us to simultaneously optimize for multiple attributes, and doesn't provide an interpretable way to split the problem into distinct subsystems.

To address these limitations, we show two methods that can be used individually or combined. The first is *cost function design*: multiple attributes may be simultaneously optimized for by adding terms to the cost function, and prioritized by varying their importance in the cost term. This allows for tuning the overall system to generate gaits that explicitly meet the multiple desired criteria. However, it doesn't allow us to guide the system internally to cleanly and distinctly create representations that capture the attributes. To address this, we demonstrate *latent space regularization*: the architecture is split into attribute-specific subsystems, each calculating targeted latent representations for a single attribute. Then these representations can be combined to solve the overall problem.

For style transfer and trajectory generation, there are no well-accepted standards for quantitatively assessing success. Instead, researchers rely on visual inspection or on minimizing squared error with examples. The problem with focusing on error is that it doesn't necessarily capture the differences that distinguish examples from different attribute classes. For example, a synthetic image of a cat might have large error from any particular example image of cats, but still would be

Manuscript received 31 December 2022; revised 25 April 2023; accepted 2 May 2023. Date of publication 8 May 2023; date of current version 17 May 2023. This work was supported in part by the NSF National Robotics Initiative Integrative under Grant 2024446/2024237. *(Corresponding author: Abhishek Sharma.)*

Abhishek Sharma is with the Department of Mechanical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: as711@uw.edu).

Eric Rombokas is with the Department of Electrical and Computer Engineering and the Department of Mechanical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: rombokas@uw.edu).

Digital Object Identifier 10.1109/TNSRE.2023.3273599

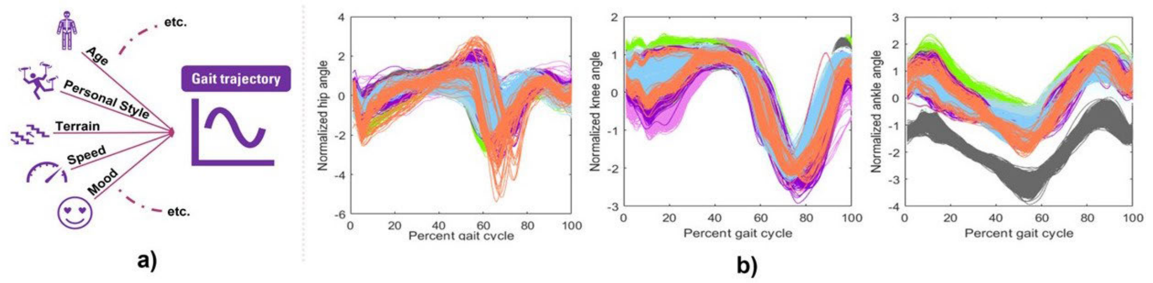

Fig. 1. (a) Multiplicity of gait attributes, (b) Gait cycles of hip, knee, and ankle angles from 7 subjects at speeds ranging from 0.5 m/s to 1.8 m/s. Different colors represent different subjects. Note that each subject shows a distinct and largely consistent strategy for movement.

recognizable as a cat and not mistaken for a dog. Therefore we propose a quantitative success measure: classifiers that identify which group an example was drawn from. If a synthetic gait trajectory is correctly classified, it is considered successful in emulating that gait attribute.

A. Multiplicity of Gait Attributes and Style Transfer

There can be multiple attributes that contribute to the generation of gait, from anthropometrics and body dimensions, to the environmental constraints, speed, to the somewhat ineffable personal style or mood of the walker. Generating gait from diverse attributes of gait can be challenging to do manually; especially for personal style which can be influenced by developmental history, injuries and other inexpressible event and thus hard to define.

We want to generate synthetic trajectories and we want them to be appropriate in multiple ways simulataneously. Individuality and speed are the most obvious examples. In the literature, there are methods for generating trajectories that are appropriate in a single way, usually in terms of low root mean square error (RMSE). What we are showing here is a general way to use representation learning for an arbitrary number of simultaneous constraints.

In machine learning, "style transfer" has been used to generate samples that meet multiple simultaneous criteria. Style transfer has primarily been used in computer vision for generating images. The content of one image can be recast into the style of another. Fig. 2 shows an application using an adversarial autoencoder [5] in rendering digits of the MNIST dataset in the style of other example digits. Each row in the figure has the same handwriting style, generated by a distinct individual. This technique was used to transfer a single attribute, dubbed to be "style", but in the current manuscript paper we transfer both style and speed. We limit our analysis to style and speed in this manuscript but the methods we present can be used to transfer an arbitrary number of distinct attributes, simply by including more attribute-specific encoders and terms in the cost functions (See section I-C).

B. Quantitative Assessment of Generated Trajectory Quality

A person's style of walking is an important aspect of gait, perhaps even the most important component of prediction error in gait models [6], yet it is hard to measure and quantify. Fig.1b shows how different even relatively simple treadmill walking can be across subjects. To assess the quality of a generated trajectory, previous methods [4] use RMSE between the generated trajectory and a single representative trajectory or an ensemble of example trajectories. If a personalization method leads to reduction in RMSE, that is considered an indication of better personalization. However, the same RMSE can be achieved by two very different pairs of trajectories, potentially very different in their personalization success. For example, a trajectory with joint angles that briefly overshoot

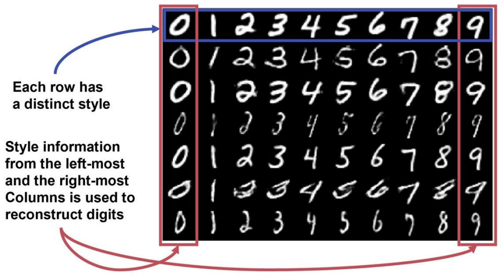
Each row has a distinct style
Style information from the left-most and the right-most Columns is used to reconstruct digits

Fig. 2. An example of style transfer in computer vision. Images of handwritten numbers (MNIST dataset) are used as example inputs. given examples of handwriting style, synthetic handwritten numbers can be generated. adapted from [5].

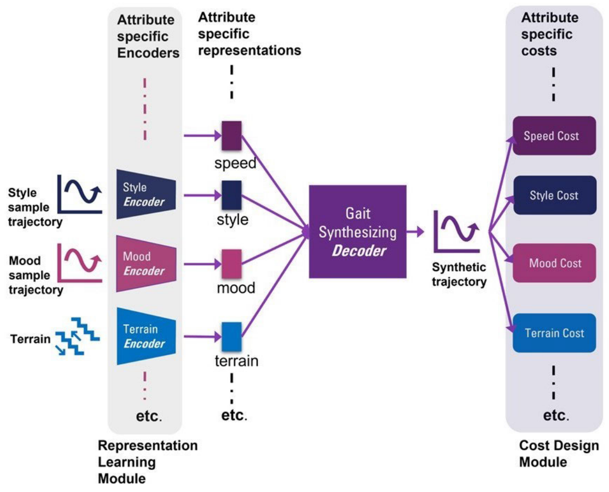

Fig. 3. Concept diagram for generating gait with multiple desired attributes. examples taken with different attributes, such as style, mood, or terrain, are encoded into attribute-specific representations. These latent spaces are learned, where that learning is encouraged to represent differences in that attribute and discouraged from representing differences in the other attributes. These representations are decoded to create synthetic trajectories. The decoder is learned using a multi-term cost function designed to encourage simultaneous transfer of each of the attributes.

and then stabilize could have the same RMSE as one with a small bias that persists. These differences are potentially quite important to the subjective experience of personalization, but are not well expressed by RMSE.

To help address this, we propose a quantitative method based on training classifiers on the example data. The success rate of the classifier can be considered the ceiling for how discriminable the classes are. If a generated trajectory is sufficiently similar to the desired person or speed, it will fool that classifier, and this can be quantitatively measured.

C. Optimizing Trajectory Synthesis via Optimal Representation Learning

The method used for handwriting style transfer (Fig. 2) can be directly adapted to generate gait, where instead of handwriting style, we synthesize walking style, and instead of which digit, we synthesize trajectories for different speeds. However, this straightforward adaptation of the method doesn't allow us to have much control over the output of the network. Firstly, minimizing RMSE doesn't necessarily lead to best person and speed appropriate performance. Secondly, the representations learned are not necessarily encoding the intended attributes (personal style in this manuscript). To address this, we split gait generation into two distinct components: learning appropriate representations, and composing the learned representation to generate trajectories. Thus optimization can be applied at both stages (Fig. 3): at the output generation stage using cost function design, and at the attribute encoding stage using latent space regularization.

1) Cost Function Design: Training gait models by minimizing RMSE doesn't necessarily promote attribute specificity in the output of the model, and might not be the best indicator of clinical suitability of the generated gait. The same RMSE can be achieved by two very different trajectories, and neural network convergence to lower RMSE doesn't necessarily mean that the solution is adequate. For example, certain phases of gait are more crucial to get right than others e.g., heel strike or toe-off. Therefore, constraining model outputs in more desirable ways by adding more attribute specific cost could be beneficial. In this work, we penalize the outputs of the model by using person and speed specific classifiers pre-trained on real trajectories from the training data. This approach could be extended to include cost terms for other desirable attributes. These cost terms can then be weighed to emphasize one attribute in the output than other. For example, a diffusion-model approach to creating human motion for animation used a custom foot contact loss term to constrain realistic foot-ground interactions [7].

2) Latent Space Regularization: Learning appropriate representations is crucial for successful machine learning systems. Not only can it help achieve better performance, but can help in interpretation of the results and in diagnosing model outputs. This is even more crucial when it comes to generative models. There are several examples in machine learning of regularization to achieve desired latent representations. For example, variational autoencoders enforce a Gaussian distribution on the bottleneck. This prevents fractured representation in the latent space and allows the decoder to create more plausible samples. Adversarial autoencoders use adversarial training to enforce any desirable distribution on the latent space. In this manuscript, we show the benefit of controlling the latent representation for the task of generating personalized gait trajectories. We do this by not only promoting the desired attribute (i.e., style) in the latent space but also penalizing it from encoding speed information. This helps achieve drastic performance gains over the other approaches that don't explicitly control the latent representation.

D. Using Classifiers in Two Distinct Ways

It should be noted that this research is using classifiers in two distinct and independent ways. The first is as a quantitative measure for person and speed specificity of the synthetic trajectories (Section I-B). These classifiers need to be able to classify real trajectories in terms of person and speed with a high degree of accuracy, and are trained on all the data from all the subjects. The second is in training: differentiable classifiers are used to regularize the latent representations or as terms in the cost function, to enforce desired attributes in generated trajectories, as described in Section I-C. These classifiers are trained using the training data only to prevent information leakage from the test data during training.

E. Personalization in Practice: How Would This Be Used?

Fig. 4 describes how the personalization process (Fig.3) could be used. Envision a lower-limb assistive device such as a prosthesis or exoskeleton, that needs to generate personalized gait at several desired speeds. Gait samples from the target user are recorded, at a few selected speeds for which the assistive device has been tuned as per current clinical practice. These trajectories are then mixed with a databank of gait samples for a wide range of individuals and speeds. The resultant dataset is used to train neural network based encoder-decoder models (See Figs. 5-7). Speed information about the input trajectories is used to disentangle speed and style information. This allows the encoder to learn to extract personal style from input trajectories. The decoder learns to compose style and speed to generate trajectories of desired style and speed. Thus, we can extract style from a few optimized examples of the target user and combine them with desired speeds to interpolate and extrapolate across a wider range of speeds.

F. Contributions

- We demonstrate that classifiers can be used as a quantitative metric, beyond the more generally used RMSE, to evaluate the person-specificity and speed-specificity of a generated gait.
- We present a style transfer based framework for personalization of assistive devices. Style is extracted by a neural network encoder from samples of human gait at known speeds and used by a decoder to generate personalized gait at a variety of desired speeds.
- We use two generalizable regularization methods, cost function design and latent space regularization, to optimize the learned representations. These methods are more interpretable and customizable than a baseline style transfer method, and yield improved synthetic trajectories.

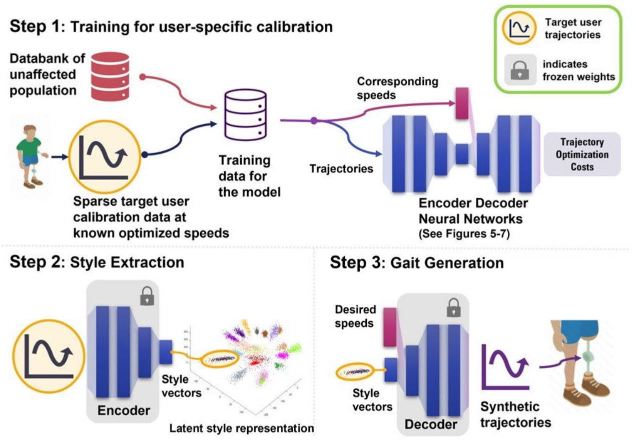
Step 1: Training for user-specific calibration
Target user trajectories
indicates frozen weights
Databank of unaffected population
Corresponding speeds
Training data for the model
Trajectories
Trajectory Optimization Costs
Sparse target user calibration data at known optimized speeds
Encoder Decoder Neural Networks (See Figures 5-7)
Step 2: Style Extraction
Step 3: Gait Generation
Encoder
Style vectors
Latent style representation
Desired speeds
Style vectors
Decoder
Synthetic trajectories

Fig. 4. How the system would be used in practice. Step 1: gait examples are recorded for a sparse set of speeds, tuned according to current clinical practice for eg. slow, medium, and fast walking. These data, plus the databank of gait examples, form the training data for the model. Training results in a model that is well calibrated to the target user. Step 2: User-specific style representations are pre-extracted to be used at run-time. Step 3: Style representations and desired speeds are input to the decoder to create a combined user-and-speed-appropriate synthetic trajectory.

II. Methods

A. Dataset

We used the publicly available dataset from [8]. The data were recorded from 22 able-bodied subjects walking at various speeds. In this manuscript, we only consider the data from treadmill walking and use the data at 14 speeds 0.5 m/s to 1.8 m/s at an increment of 0.1 m/s. The resultant dataset has a total of 9073 trajectories. We selected 2 subjects at random as target users, and use the data from the remaining 20 subjects to form the "databank of unaffected population" (See Fig. 4).

B. Data Processing

The right hip, knee and ankle kinematics data at desired speeds was extracted from the MATLAB files available on Camargo et al. The kinematics data was segmented into gait cycles and time normalized to 100 points. Thus each gait sample is in a $R^{100\times3}$ space, where 100 is the number of time points and 3 is the number of joints. Each joint is normalized using its mean and standard deviation across all the training data. Each input gait sample is flattened into a R^{300} vector, for the neural networks to process.

C. Analysis

1) Root Mean Square Error: We use the commonly used root mean square error (RMSE) to measure the distance (in degrees) between the real and generated trajectories. We report the mean of pairwise RMSE between the real and the generated trajectories, and compare it with the previous gait generation methods. (1), as shown at the bottom of the next page.

2) Person Specificity and Speed Specificity: As described in Section I-B, we use classifiers to quantify the quality of generated trajectories. If the generated trajectories can fool classifiers trained on real trajectories, then the generator can be considered successful. The strongest example of this would be if the generator can fool classifiers from a different model class. Since we use neural network classifiers within the generator, (see Section II-E) we first need to see how successful non-neural-network classifiers can be. These classifiers (see Table II) were trained on all the ground truth data (both training and test sets), validated using 5-fold cross-validation. We chose a set of well-established classifier types whose behavior is well understood, and are arguably more interpretable and can be vetted by clinicians. We trained the following 5 model types- Decision Trees, Linear Discriminant, Naive Bayes (NB), Support Vector Machines (SVM) and K-Nearest Neighbor (KNN) using the MATLAB 2020b classification learner toolbox, to separately classify real trajectories in terms of person and speeds. The hyperparameters corresponding to the best model for each model type are listed in Table I The SVM models perform the best with person classification accuracy of 99.8% and speed classification accuracy of 83.9%. Table II shows the speed and person classification performance for the best model in each model type.

The second purpose of the classifiers is to guide training. *Neural Network* classifiers are used to regularize latent spaces and design cost functions, as described in section II-E. Using Neural Networks classfiers allows differentiable layers through which gradients can be computed and backpropagated, and can be easily integrated with the Encoder-Decoder backbone see Fig. 5. These classifiers are trained on only the real trajectories from the training data to prevent leakage from the test data. The classifiers are validated using 5-fold cross validation to ensure robustness in hyperparameter selection.

D. Baseline Encoder-Decoder Architecture

Based on the concept described in Fig. 3 and the supervised adversarial autoencoder used for hand writing style transfer on the MNIST dataset in [5], we use an encoder-decoder architecture. The encoder extracts the information independent of speed i.e., person-specific style. The decoder synthesizes the trajectories by combining the style information in 'z', with the desired speed information injected using one-hot encoding in 'y'. The model is trained by reconstructing the input trajectories with RMSE as the loss function. This is different from the MNIST style transfer in that the output for each dimension in the MNIST case is binary and hence the task can be framed as pixel-wise classification problem, while here the task is fundamentally a regression problem due to continuous trajectories. This model serves as our baseline for performance and we introduce regularization on top of this architecture in the next section to improve over the baseline.

1) Encoder Details: The architecture of the baseline encoder-decoder is shown in Fig. 5. The encoder has 4 layers, which transform R^{300} input trajectories to their R^3 latent style representation. Latent style space is chosen to be 3-D to make visualization easy. The first 2 layers of the encoder have *leaky-ReLU* activation with a slope of 0.1 for negative values.

TABLE I
HYPERPARAMETERS FOR PERSON AND SPEED CLASSIFIER

Models	**Person classifier**	**Speed classifier**
Decision Trees	Preset: Fine Tree; max splits = 100; Split criterion: Gini's diversity index	Preset: Fine Tree; max splits = 100; Split criterion: Gini's diversity index
Linear Discriminant	Covariance structure: Full	Covariance structure: Full
Naive Bayes (NB)	Preset: Kernel Naive Bayes; Distribution: Kernel smoothing; Kernel: Gaussian; Support: Unbounded	Preset: Kernel Naive Bayes; Distribution: Kernel smoothing; Kernel: Gaussian; Support: Unbounded
Support Vector Machine	Kernel: Quadratic; Kernel scale: Automatic; Box constraint level: 1; Multiclass method: One-vs-One; Standardize data: true	Kernel: Cubic; Kernel scale: Automatic; Box constraint level: 1; Multiclass method: One-vs-One; Standardize data: true
K Nearest Neighbors	Preset: Weighted KNN; Number of neighbors = 10; Distance metric: Euclidean; Distance weight: Squared inverse; Standardize data: true	Preset: Weighted KNN; Number of neighbors = 10; Distance metric: Euclidean; Distance weight: Squared inverse; Standardize data: true

TABLE II
PERSON AND SPEED CLASSIFICATION ACCURACY ON REAL TRAJECTORIES FOR DIFFERENT CLASSIFIERS

Accuracy	**Tree**	**LDA**	**NB**	**SVM**	**KNN**
Person	91.8 %	99.5 %	98.2 %	**99.8** %	99.5 %
Speed	38.1 %	52.5 %	40.6 %	**83.9** %	76.6 %

The following two layers use linear activation. We do not present an exhaustive search of architecture choices here; the presented architecture generally resembles our previous successful learned latent representations for human movement [9], [10].

2) Decoder Details: The decoder (Fig. 5) takes the style representation of the input trajectory concatenated with one-hot representation of the corresponding speed, and uses it to reconstruct the input trajectory. The decoder has 4 layers, with the first 3 layers using *leaky-ReLU* activation functions with a slope of 0.1 for negative values. The final layer uses a linear activation function. A root-mean-squared reconstruction error loss is minimized to reduce the distance between the real and generated trajectories. (Eq. 2)

$$\mathcal{L}_{total} = \mathcal{L}_{recon} \tag{2}$$

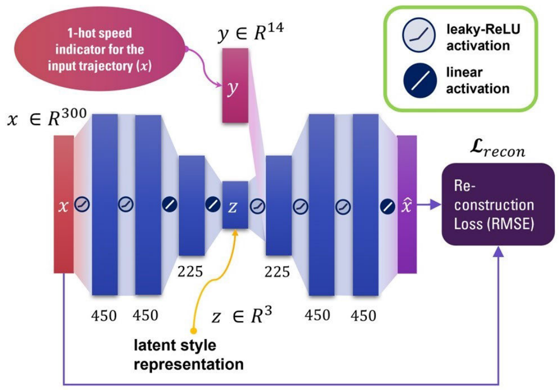

Fig. 5. Baseline style transfer architecture, based on [5]. The encoder extracts person specific style from the input trajectories and the decoder uses the style information and desire speed concatenated at the latent space, to generate person and speed specific trajectory.

E. Gait Generation Using Latent Space Regularization and Cost Function Design

The baseline encoder-decoder architecture uses speed injection at the decoder input to encourage the encoder to learn the complementary information i.e., personal style. This however is contingent on factors like the size of latent space 'z', and doesn't allow much control over what information is preserved in 'z'. This also doesn't allow us to control how the decoder combines 'y' and 'z' to generate the trajectory. Thus, we introduce two additional changes: 1) cost function design at the output of the decoder to emphasise the desired attributes in the synthesized trajectories, and 2) latent space regularization to control what information is encoded in the 'z' space.

1) Output Regularization Using Cost Function Design: We used person and speed classifiers, trained on real trajectories from the training set, to regularize the model at the output via terms in the loss function. The classifiers take the trajectories

$$mean\ pairwise\ RMSE = \frac{\sum_i \sum_j \sum_{n_{ij}} \sum_{m_{ij}} RMSE(x_{m_{ij}}, \hat{x}_{n_{ij}})}{\sum_i \sum_j \sum_{n_{ij}} \sum_{m_{ij}} 1}$$
$$i = \text{person index}, \quad \text{j = speed index}$$
$$m_{ij} = \text{real trajectory index for person i and speed j}$$
$$n_{ij} = \text{generated trajectory index for person i and speed j}$$
$$x_{m_{ij}} = m_{ij}^{th} \text{real trajectory for person i and speed j}$$
$$\hat{x}_{n_{ij}} = n_{ij}^{th} \text{generated trajectory for person i and speed j} \tag{1}$$

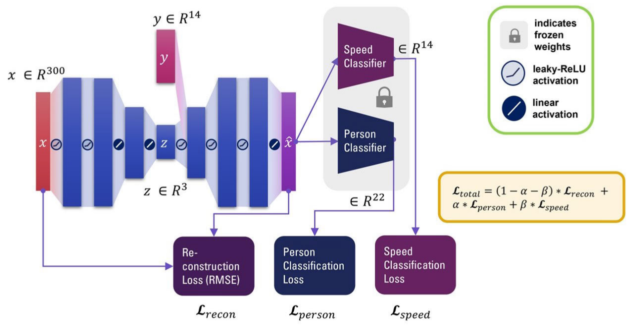

Fig. 6. Output Regularization using cost function design. In addition to the reconstruction loss, classifier losses are added, weighted by coefficients $\alpha$and β.

generated by the decoder as input, and output the person and speed classification losses, which are weighed by corresponding parameters α and β, and added to the reconstruction RMSE loss. Grid search over α and β is used to find the optimal weights. These classifiers enforce desirable traits i.e., person-specificity and speed-specificity in the outputs from the decoder. In this way, we are directly rewarding the system for achieving our goal - to creating synthetic trajectories that resemble the person or the speed, beyond RMSE. This approach is depicted in Fig. 6.

$$\mathcal{L}_{total} = (1-\alpha-\beta) * \mathcal{L}_{recon} + \alpha * \mathcal{L}_{person} + \beta * \mathcal{L}_{speed} \tag{3}$$

2) Latent Space Regularization: Using classifiers at the output doesn't allow us to explicitly control the attributes encoded in the latent space 'z'. The designer's control is only exerted at the cost function, but it's up to the learning algorithm to organize how the latent space is encoded and decoded. This can be difficult to interpret, and doesn't provide a detailed way to encourage distinct parts of the architecture to take on distinct purposes. To address this, we use latent space regularization to directly force the latent space to encode person-specific information, while penalizing the inclusion of speed-specific information (See Fig. 7). Latent regularization is done in two steps: First, the encoder is trained to take real trajectories as input and produce a person classification. A negative speed classification cost is added to remove speed information in 'z'. These cost functions are weighed by a parameter 'γ', which is tuned using grid search. In the second step, after the encoder has been completely trained, the decoder portion of the architecture is trained to minimize reconstruction loss, with the encoder weights frozen. This prevents encoder representations learned in step 1 from changing. This approach is depicted in Fig. 7.

$$\begin{aligned}\mathcal{L}_{step\ 1} &= (1-\gamma) * \mathcal{L}_{person(latent)} - \gamma * \mathcal{L}_{speed(latent)} \\ \mathcal{L}_{step\ 2} &= \mathcal{L}_{recon}\end{aligned} \tag{4}$$

3) Combining Output and Latent Regularizations: Output regularization helps us emphasize desired attributes in the output of the decoder, while latent space regularization gives us explicit control over attribute specific representations learned in the latent space 'z' and therefore what information is available to the decoder for trajectory synthesis. These methods

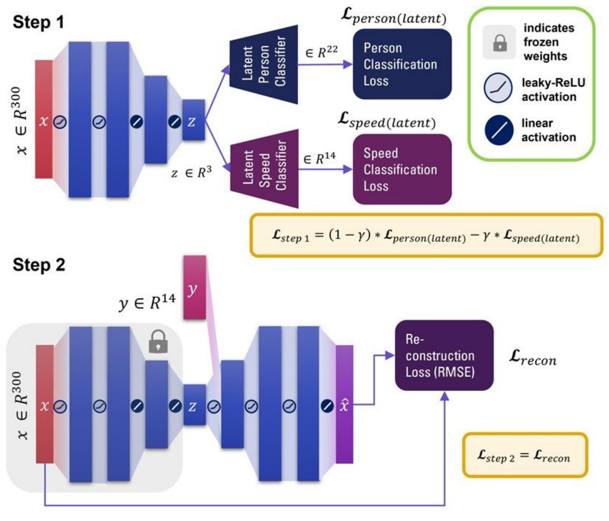
Step 1
$x \in R^{300}$
$z \in R^3$
Latent Person Classifier
$\in R^{22}$
$\mathcal{L}_{person(latent)}$
Person Classification Loss
Latent Speed Classifier
$\in R^{14}$
$\mathcal{L}_{speed(latent)}$
Speed Classification Loss
indicates frozen weights
leaky-ReLU activation
linear activation
$\mathcal{L}_{step\ 1} = (1-\gamma) * \mathcal{L}_{person(latent)} - \gamma * \mathcal{L}_{speed(latent)}$
Step 2
$y \in R^{14}$
$x \in R^{300}$
Re-construction Loss (RMSE)
$\mathcal{L}_{recon}$
$\mathcal{L}_{step\ 2} = \mathcal{L}_{recon}$

Fig. 7. Latent Space Regularization. The encoder is trained first. Latent representations are regularized by classifiers in the latent space. The person classifier should be successful, indicating that latent representations of individuals are separated in the latent space. Speed classification, on the other hand, should fail, indicating that speed information is not encoded into the latent space. This is reflected by the negative coefficient for the speed loss term. The trained encoder weights are fixed, and the decoder is trained as in the baseline according to reconstruction loss.

have mutually exclusive advantages and combining them could lead to overall better performance. Thus, we present a model that combine the two methods. The combined model is trained by first pretraining the encoder similar to latent regularization (Fig.7; Step 1). Then, the entire model is trained like step 2 with person speed classification cost applied to the output of the decoder.

$$\begin{aligned}\mathcal{L}_{step\ 1} &= (1-\gamma) * \mathcal{L}_{person(latent)} - \gamma * \mathcal{L}_{speed(latent)} \\ \mathcal{L}_{step\ 2} &= (1-\alpha-\beta) * \mathcal{L}_{recon} + \alpha * \mathcal{L}_{person} + \beta * \mathcal{L}_{speed}\end{aligned} \tag{5}$$

F. Calibration Using Target User Examples

Everyone has a unique walking *"style"* which depends on several factors like height, weight, medical history etc. Thus, we require a few sample trajectories from the target user, to extract style representations and calibrate our models to the target users.. These 'calibration samples' can be obtained by capturing target user gait at a few speeds. In the case of prostheses/orthoses users these trajectories can be obtained by handtuning as per the standard clinical practices i.e., tuned trajectories at selected slow, medium and fast speeds. Then, trajectories at rest of the speeds can be interpolated or extrapolated.

For comparison among baseline, output regularization, latent regularization, and combined, we used sample trajectories at 3 speeds: slow (0.7 m/s), medium (1.1 m/s), and fast (1.5 m/s), mixed with trajectories from a databank from subjects with no mobility impairments, for training. Three example speeds were chosen because it is a reasonable amount of data to expect a user to provide during a single visit. The three speeds span from the slowest to the fastest speeds to capture the maximal variations due to speed. However, we wanted

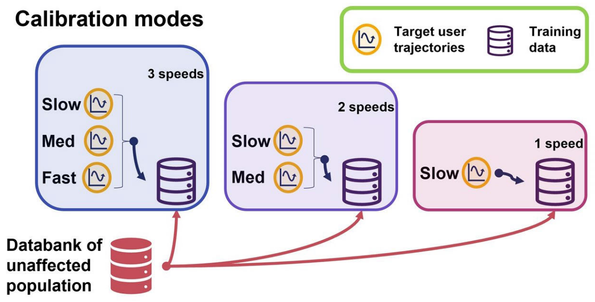

Fig. 8. Calibration modes: Easy mode is where trajectories at 3 speeds from target users are included in the training data. Moderate mode includes 2 speeds while the hard mode only includes slow speed.

TABLE III
OPTIMAL HYPERPARAMETERS FOR BASELINE, AND OUTPUT REGULARIZATION. NETWORK WERE TRAINED FOR 10000 EPOCHS

Hyperparam.	**search range**	**baseline**	**output reg.**
Learning rate	$[10^{-6} : 10^{-1} : 10^{-2}]$	10^{-5}	10^{-4}
Person cost weight (α)	$[0.0 : 0.1 : 0.5]$	n/a (0.0)	0.4
Speed cost weight (β)	$[0.0 : 0.1 : 0.5]$	n/a (0.0)	0.4

to determine how sensitive the results are to this choice. Therefore, we have conducted another experiment using fewer example speeds. The best method from the above comparison was chosen (combined, see Section III) and trained using fewer speeds from the target user data training data (Fig. 8). The easiest mode for the system has the most examples trajectories, at 3 speeds (slow, medium, and fast) from the target user. The moderate mode has only 2 speeds (slow and medium), and the hardest mode has only 1 speed (slow), included in the training data.

G. Hyperparameter Tuning

We compared 4 methods in this manuscript, all using variations on the encoder-decoder architecture. The tuned hyperparameters were learning rates, person cost weight (α), speed cost weight (β) and latent speed cost weight (γ). Grid search was used to find the optimal hyperparameters. A random selection of 20% samples from the training data is used as validation set for optimal hyperparameter selection (not used for model weights updates). Baseline and output regularization are trained in a single step. The range of hyperparameter values tested and the optimal values for both the methods are shown in Table III. Latent regularization and combined regularization are trained in 2 steps. The first step trains the encoder and the second step trains the decoder. Table IV shows the corresponding hyperparameters.

In the original work with hand writing style transfer, [5] regularization using an adversarial network was used as well, to force enforce a Gaussian distribution in the latent space. In our experiments we found that using adversarial network made training difficult and reduced the overall performance for the baseline model in terms of the key RMSE metric. Therefore, we set the learning rate for the adversarial update to 0 for all the experiments. Thus, the Gaussian distribution was not enforced on the latent space.

TABLE IV
OPTIMAL HYPERPARAMETERS FOR LATENT REGULARIZATION, AND COMBINED REGULARIZATION. EPOCHS FOR ENCODER AND DECODER TRAINING ARE 10000 AND 5000 RESPECTIVELY

Hyperparam.	**search range**	**latent reg.**	**combined**
Encoder learning rate	$[10^{-5} : 10^{-1} : 10^{-3}]$	10^{-5}	10^{-5}
Decoder learning rate	$[10^{-5} : 10^{-1} : 10^{-2}]$	10^{-4}	10^{-4}
Latent speed cost weight (γ)	$[10^{-13} : 10^{-1} : 10^{-7}]$	10^{-10}	10^{-10}
Person cost weight (α)	$[0.0 : 0.1 : 0.5]$	n/a (0.0)	0.2
Speed cost weight (β)	$[0.0 : 0.1 : 0.5]$	n/a (0.0)	0.2

For step 1 of latent regularization, the use of negative weight for the speed classification loss sometimes led to exponential blowup of the speed cost term, effectively stagnating improvements in person loss. We found that very small cost weights were required to train the encoder effectively (Table IV). For the combined method, we used the encoder from step 1 of latent regularization.

III. RESULTS AND DISCUSSION

Fig. 9 shows the mean trajectories generated by each method superimposed with the the mean real trajectories of a randomly drawn target user, for all the speed. Table V shows the performance of different methods in terms of RMSE, person classification accuracy and speed classification accuracy. The leftmost column of Table V shows pairwise RMSE between two real trajectories of same speed and person. This indicates the inherent variability of the real trajectories, even for a given person and speed. The baseline model, which we adapted from handwriting style transfer [5], was capable of generating trajectories with RMSE of 1.29, 2.87 and 1.78 degrees for hip, knee and ankle respectively. For comparison, the method shown in [1] show RMSE values of 2.31, 3.46 and 2.60 degrees respectively. Thus, the baseline model performs slightly better in terms of RMSE. They personalize across speeds (albeit a smaller range) as well as incline as compared to speed only in our case, which might explain a higher RMSE.

However, the trajectories generated by the baseline method show relatively low performance for person and speed appropriateness. Even though the RMSEs are similar (Table V, first 3 rows), the classification success is quite poor (Table V, bottom two rows.) The trajectories are not capturing the unique features that make them person and speed specific. This supports our initial guess that RMSE might not necessarily capture other attributes like personal style and speed. This also provides evidence that there is a need to use other metrics to assess our models, similar to how we use the SVM person and speed classifiers here.

The baseline style transfer method also doesn't provide explicit control over attributes of generated trajectories. For example, if an application prioritized accurate person specificity, with less importance placed on speed, there is no straightforward way to adjust the method accordingly. As shown in Fig. 5, the decoder simply takes in the one-hot vector y representing the desired speed, and combines that with the latent representation of personal style, z, to output a trajectory that is penalized for reconstruction loss. In exploratory experiments, we varied the number of dimensions of the latent representation z. We found that a low number of dimensions would lead to better speed performance while reducing person performance, while a higher number of dimensions would result in better person performance and bad speed performance.

A. Regularizations

Neural network classifiers trained only on the training data were used to penalize speed and person performance, allowing us to emphasize these desired attributes in the generated trajectories, along with low RMSE, by designing a cost function including classifier loss terms. Table V shows that output regularization leads to performance improvement over the baseline in terms of speed and person classification as well as RMSE. While this enhances overall performance, it doesn't allow us to control the representations learned in the latent space 'z'. To do this, we regularized the latent space explicitly, which also results in overall performance improvement over the baseline. The performance improvement is drastic in terms of person specificity, but not as great for speed specificity. This is in contrast to output regularization, which shows drastic improvement in terms of speed-specificity.

By combining the two complementary approaches, we get a system that provides almost as successful person classification, and much improved speed classification (Table V Combined, rightmost column). So we conclude that for this data, the combined strategy is the overall best if one cares about both person and speed classification simultaneously.

B. Latent Space Visualization

Using bottlenecked, low-dimensional, latent spaces is not only advantageous in terms of overall model performance, but also allows for visualization or manipulation of the encoded data in a tractable space. Fig. 10 shows the latent style representations learned by the baseline model (left) and the two regularization methods (center and right) from 2 viewpoints each. The latent space of the combined approach is the same as the latent regularization only, because the same encoders were used. Each subject is assigned a unique color. The orientation of axes was chosen so that the target users are clearly visible for each method. We see that output regularization doesn't greatly change the pattern and arrangement of subjects in the latent space from the baseline method or make the subjects more separable. It does, however, create better clusters for the target users, albeit not completely separable from the other users. Latent regularization distinctly makes tighter clusters, with easily separable subjects. It also tends to push subjects far apart from each other in the radial direction. This indicates the usefulness of pretraining the attribute specific encoder (here style encoder) with emphasis on removing information related to other attributes (here speed). This creates more interpretable and unadulterated representations, which can then be used for various downstream tasks, e.g., personalization. In this manuscript we used these representations for a specific task i.e., to generate gait trajectories at a few desired speeds. However, these representations can potentially be used to personalize other predictive models of gait [11], [12], [13], instead of using more standard quantifiable attributes like height, weight etc.

C. Amount of Calibration Data

We performed a sensitivity analysis to understand how the results would degrade given fewer examples. Other methods like [1], [3], and [14] have also been shown to work with a single sample. In this analysis, we use two other metrics i.e., person and speed classification, in addition to RMSE which is used by other studies. Thus, We chose two additional relatively challenging scenarios: a moderate version where the user provides walking examples at 0.7 m/s and 1.1 m/s, and a hard version where the user provides walking examples only at 0.7 m/s. For all three, the combined latent and output regularization strategy was used. As can be seen in Table VI, performance degradation is quite small. We see a worst-case reduction of 1.03% in person classification accuracy and even a 0.13% increase in speed classification accuracy. Considering the variability of trajectories (See Table V; leftmost column), this appears to be functionally equivalent. From this experiment we infer that a single example speed should be sufficient to use the proposed system. This is a significant result since this has the potential to greatly reduce the amount of effort required to tune prostheses controllers, which currently limits their capabilities.

D. Beyond Personalization

In this manuscript we showed an application of the concept illustrated in Fig. 3, where we chose personalization of generated gait at different speeds. However, the methods are application agnostic. With the recent rise in multi-modal gait datasets [8], [15], [16], [17], [18], [19], [20], we can also learn representations of hard to define factors in a similar way the representations of style were learned in this manuscript. Similarly, we could encode terrain in an appropriate latent space and then compose different representations using the decoder. As we have shown here, these approaches to representation learning are compatible with optimization of the decoder by including attribute specific cost terms, and these can be related to the performance of classifiers. This means that we don't necessarily have to have first-principles understanding of what makes the classes different, if we can simply sample many examples from them.

E. Limitations

This work does not present any analysis of the clinical effects of personalized prosthesis control. Whether a more personalized trajectory would lead to any improvements in the quality of life of the target users stands to be tested and should be investigated in future works. The methods presented here depend on the assumption that the user can walk optimally with at least one speed. Thus, the method relies on the clinician's expertise to tune the device at the calibration

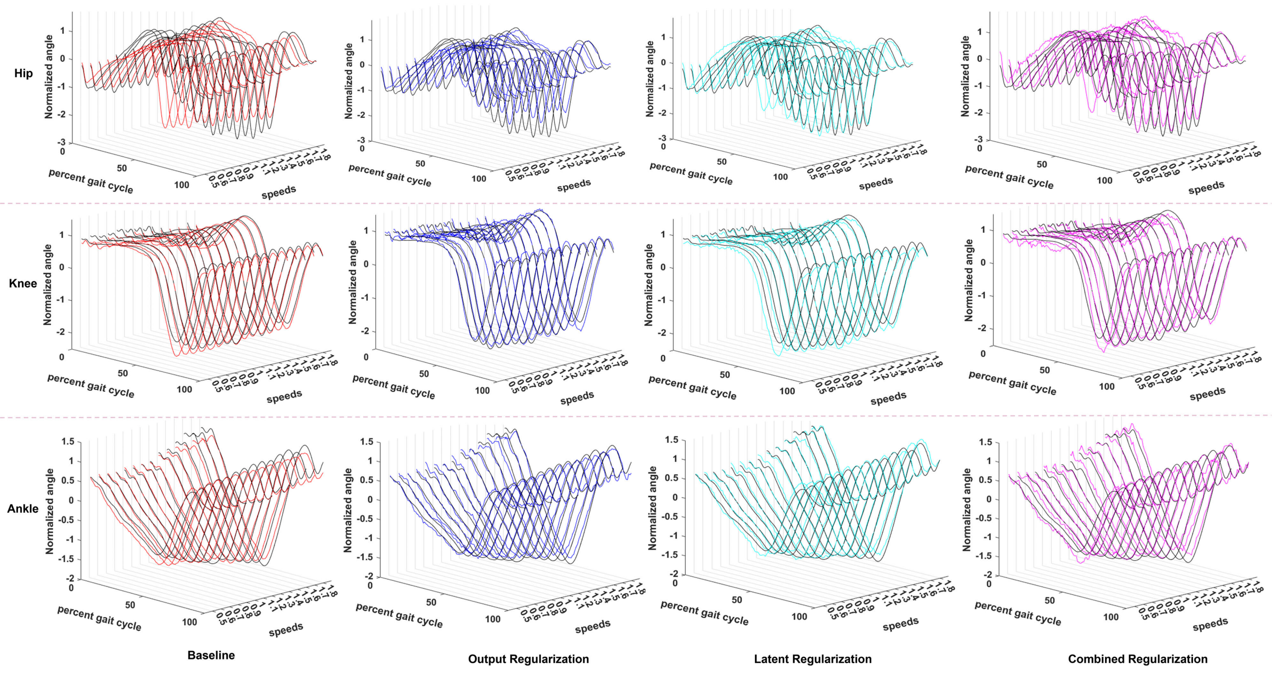

Fig. 9. The mean generated trajectories for Hip (top), Knee (middle), Ankle (bottom) of one target user are shown, for the 4 methods: Baseline (red), Output Regularization (blue), Latent Regularization (Cyan), and Combined Regularization (Magenta). The trajectories are superimposed on the mean real trajectories (shown in black). We see that the regularization method capture the changes in speed better compared to the baseline, more so for the hip and knee joints.

TABLE V

Comparison of the Baseline and Regularization Methods. The Leftmost Column Shows the Mean of RMSE Between Each Pair of Real Trajectory From the Same Class (Person and Speed), and the Performance of Person and Speed Classifiers on Real Trajectories (See Table II). Number of Target User Samples and Generated Trajectories 166 and 2324, Respectively

Performance	Real	Baseline	Output Reg.	Latent Reg.	Combined
Hip mean RMSE (deg)	1.03	1.29	1.18	**1.08**	*1.12*
Knee mean RMSE (deg)	2.30	2.87	2.60	**2.45**	*2.54*
Ankle mean RMSE (deg)	1.48	1.78	1.63	**1.49**	*1.57*
Person Classification	99.8 %	81.67 %	89.37 %	**99.78** %	*99.01 %*
Speed Classification	83.9 %	30.81 %	**80.16** %	49.31 %	*64.16 %*

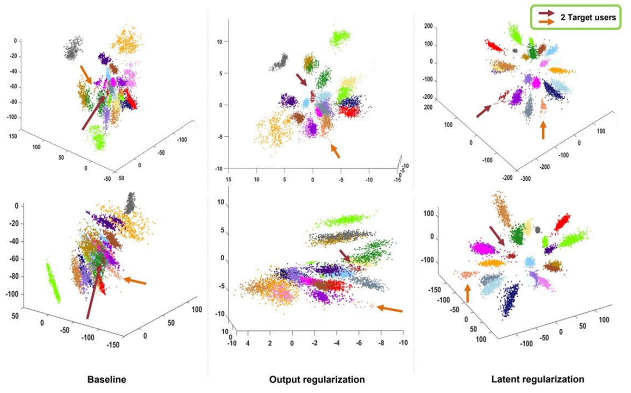

Fig. 10. Latent space visualization: Each row shows two different views of latent encoding of the trajectories from all the subjects, for baseline, output regularization and latent regularization methods. The encoding for the combined regularization is same as that for latent regularization. Different colors correspond to different subjects The two arrows indicate the target users.

speeds. Also, while we present person and speed classification as additional metrics, the final evaluation about suitability of the trajectories to a user should still be made by a clinician.

The total amount of data available for learning models was relatively small, 9073 trajectories, compared to millions of samples that are present in some datasets. Even [5], upon which the baseline model was based, used the MNIST dataset which is comprised of over 60,000 samples. Thus, it remains to be tested how cost function design compares with latent space regularization when using dramatically larger datasets. Regardless, for small datasets this manuscript shows the strengths and weakness of the different methods.

TABLE VI

Performance for the Sensitivity Analysis to the Number of Example Speeds Given

	3 speeds	2 speeds	1 speed
# target user samples	166	102	46
# generated trajectories	2324	1428	644
Hip mean RMSE (deg)	1.12	1.12	1.13
Knee mean RMSE (deg)	2.54	2.54	2.56
Ankle mean RMSE (deg)	1.57	1.58	1.59
Person Classification	99.01 %	98.74 %	97.98 %
Speed Classification	64.16 %	64.35 %	64.29 %

In this work, we represented 14 speeds using 14-dimensional 1-hot encodings, similar to how [5] used digits, but speed being a continuous signal arguably can be best represented as a single continuous variable. The effect of this choice can be invested in the future work.

Using interpretable classifiers for quantification of attribute-specific performance (speed or style) is ideal for communicating the trustworthiness of the generated trajectories to stakeholders. In this work, we avoid using neural network classifiers to evaluate the generated trajectories. Instead, we used SVMs with quadratic and cubic kernels for person and speed classifiers, since they yielded the best classification performance for real trajectories. However, more interpretable methods like KNN [21], which also perform good enough (see Table II) can be used as well. In this manuscript we don't present an exhaustive analysis of how the cost function weights α, β and γ influence the performance of the models. We only presented the best performance achieved by each model in a sparse grid search. However, understanding the behaviour of model under interaction of different cost terms can be useful for downplaying or emphasizing an attribute over the other.

F. Future Work

A study with users using a device controlled to achieve these trajectories would be necessary to understand the effects of personalization. We have limited the scope of the manuscript to generating the trajectories, but what trajectory variations are important, and what properties result in clinical meaningfulness, remains to be shown.

We used person classification along with negative cost on speed classification to regularize the latent space. This is a variation on contrastive learning [22], which is a deep and active topic in modern representation learning. In contrastive learning, samples from different classes are explicitly pushed apart from each other in the latent space, not necessarily by using classifier loss. Examples from the same classes can be pushed closer to one another as well. This is similar in spirit to what we present here and would likely have different trade-offs that would be potentially fruitful to explore.

We do not present a rigorous search over the architecture size and type. However, we anticipate that convolutional neural networks (CNN) or recurrent neural networks (RNN) instead of the fully connected network could further improve trajectory generation performance. This is because the real trajectories have a local temporal continuity built into them, and the the fully connected network generates each dimension (out of the 300 dimensions here) independently. CNN or RNN based encoders and decoders could emphasize this time-continuous aspect of the trajectories.

Conflict of Interest Statement

The authors declare no conflict of interest.

Author Contributions

Abhishek Sharma conceived the experiments, developed the algorithms and code for data processing and machine learning, created data and results visualizations, and contributed to manuscript writing. Eric Rombokas supervised this research, provided scientific guidance, and contributed to manuscript writing.

References

[1] E. Reznick, K. Embry, and R. D. Gregg, "Predicting individualized joint kinematics over a continuous range of slopes and speeds," in *Proc. 8th IEEE RAS/EMBS Int. Conf. Biomed. Robot. Biomechatronics (BioRob)*, Nov. 2020, pp. 666–672.

[2] C. Zou, R. Huang, H. Cheng, and J. Qiu, "Learning gait models with varying walking speeds," *IEEE Robot. Autom. Lett.*, vol. 6, no. 1, pp. 183–190, Jan. 2021.

[3] Y. Huang, H. An, H. Ma, and Q. Wei, "Modeling and individualizing continuous joint kinematics using Gaussian process enhanced Fourier series," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 31, pp. 779–788, 2022.

[4] E. Reznick, C. G. Welker, and R. D. Gregg, "Predicting individualized joint kinematics over continuous variations of walking, running, and stair climbing," *IEEE Open J. Eng. Med. Biol.*, vol. 3, pp. 211–217, 2022.

[5] A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey, "Adversarial autoencoders," 2015, *arXiv:1511.05644*.

[6] K. R. Embry and R. D. Gregg, "Analysis of continuously varying kinematics for prosthetic leg control applications," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 29, pp. 262–272, 2021.

[7] G. Tevet, S. Raab, B. Gordon, Y. Shafir, D. Cohen-Or, and A. H. Bermano, "Human motion diffusion model," 2022, *arXiv:2209.14916*.

[8] J. Camargo, A. Ramanathan, W. Flanagan, and A. Young, "A comprehensive, open-source dataset of lower limb biomechanics in multiple conditions of stairs, ramps, and level-ground ambulation and transitions," *J. Biomechanics*, vol. 119, Apr. 2021, Art. no. 110320.

[9] A. A. Portnova-Fahreeva, F. Rizzoglio, I. Nisky, M. Casadio, F. A. Mussa-Ivaldi, and E. Rombokas, "Linear and non-linear dimensionality-reduction techniques on full hand kinematics," *Frontiers Bioeng. Biotechnol.*, vol. 8, p. 429, May 2020.

[10] D. Boe et al., "Dimensionality reduction of human gait for prosthetic control," *Frontiers Bioeng. Biotechnol.*, vol. 9, p. 925, Oct. 2021.

[11] V. Rai, A. Sharma, and E. Rombokas, "Mode-free control of prosthetic lower limbs," in *Proc. Int. Symp. Med. Robot. (ISMR)*, Apr. 2019, pp. 1–7.

[12] V. Rai, A. Sharma, D. Boe, P. Preechayasomboon, and E. Rombokas, "Continuous and unified modeling of joint kinematics for multiple activities," *IEEE Access*, vol. 10, pp. 47509–47523, 2022.

[13] A. Sharma and E. Rombokas, "Improving IMU-based prediction of lower limb kinematics in natural environments using egocentric optical flow," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 30, pp. 699–708, 2022.

[14] R. B. D. Joshi and D. Joshi, "MoveNet: A deep neural network for joint profile prediction across variable walking speeds and slopes," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–11, 2021.

[15] C. Schreiber and F. Moissenet, "A multimodal dataset of human gait at different walking speeds established on injury-free adult participants," *Sci. Data*, vol. 6, no. 1, pp. 1–7, Jul. 2019.

[16] A. Sie et al., "Descending 13 real world steps: A dataset and analysis of stair descent," *Gait Posture*, vol. 92, pp. 383–393, Feb. 2022.

[17] M. Palermo, J. M. Lopes, J. André, A. C. Matias, J. Cerqueira, and C. P. Santos, "A multi-camera and multimodal dataset for posture and gait analysis," *Sci. Data*, vol. 9, no. 1, pp. 1–11, Oct. 2022.

[18] G. Santos, M. Wanderley, T. Tavares, and A. Rocha, "A multi-sensor human gait dataset captured through an optical system and inertial measurement units," *Sci. Data*, vol. 9, no. 1, pp. 1–10, Sep. 2022.

[19] V. Losing and M. Hasenjäger, "A multi-modal gait database of natural everyday-walk in an urban environment," *Sci. Data*, vol. 9, no. 1, pp. 1–12, Aug. 2022.

[20] A. Sharma et al., "A non-laboratory gait dataset of full body kinematics and egocentric vision," *Sci. Data*, vol. 10, no. 1, pp. 1–11, Jan. 2023.

[21] C. Molnar, *Interpretable Machine Learning*. Morrisville, NC, USA: Lulu. com, 2020.

[22] P. Khosla et al., "Supervised contrastive learning," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 18661–18673.